PAIR: BRIDGING PERCEPTION AND ACTION IN VISION–LANGUAGE–ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models map visual observations and language instructions to continuous robot actions. This task requires a transition from representations that describe the scene and instruction to representations that support action generation. Many continuous-action VLAs leave this transition implicit and supervise it mainly through the final action-prediction loss. We introduce **PAIR**, a framework that learns a shared perception–action representation between these two spaces. During training, a Masked Action Autoencoder encodes expert action chunks into horizon-aligned Action Latent Tokens. A Bridge Module extracts task-relevant features from the current visual–language representations. PAIR aligns these features with the Action Latent Tokens to form Bridge Tokens that preserve task information and capture the structure of expert actions. The Bridge Tokens are then projected into the action-token space and injected into the initial Action Tokens, providing an action-ready starting point for Action Expert refinement. At inference, the autoencoder is removed, and the Bridge Tokens are generated only from the current observation and instruction. Experiments on LIBERO, LIBERO-Plus, and CALVIN ABC→D show gains for the evaluated OpenVLA-OFT and VLA-Adapter models. On LIBERO-Plus, PAIR raises VLA-Adapter's success rate from 59.1% to 64.2%. On CALVIN, it increases VLA-Adapter's average completed sequence length from 4.42 to 4.53. Across seven real-world tasks, PAIR raises OpenVLA-OFT's success rate from 51.4% to 65.0%. Representation analyses show that Bridge Tokens retain task information while making continuous-action information accessible before Action Expert refinement. These results support a shared intermediate representation as a useful interface between perception and action in continuous-action VLAs.

1 INTRODUCTION

Vision-language-action (VLA) policies have emerged as a promising framework for robot control. By mapping visual observations and language instructions to continuous actions, they unify visual grounding, language understanding, and control within a single policy. Recent models have achieved strong performance across diverse manipulation tasks (Brohan et al., 2022; 2023; Kim et al., 2025b; Wang et al., 2026). Yet this unified formulation hides a difficult representational transition.

The visual-language perception space and the action space are organized around fundamentally different semantics. The former describes the current world and the intended task, including scene content, object relations, and instruction-level intent. The latter specifies how the robot should act through temporally ordered, executable control. We refer to this semantic mismatch as the *perception–action representation gap*.

Existing VLA policies learn to traverse this gap primarily through end-to-end action supervision. The final action loss specifies which action should be produced, while the representation that connects visual-language semantics to action semantics is left implicit. Action tokens must therefore discover within the policy both what information is relevant to control and how it should be reorganized into an action-ready representation.

Recent work has sought to facilitate this transition by exposing control-relevant structure through visual traces, affordances, visual reasoning, and learned latent actions (Zheng et al., 2025; Nasiriany et al., 2024; Zhao et al., 2025; Liu et al., 2026b). These efforts show that making such structure

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

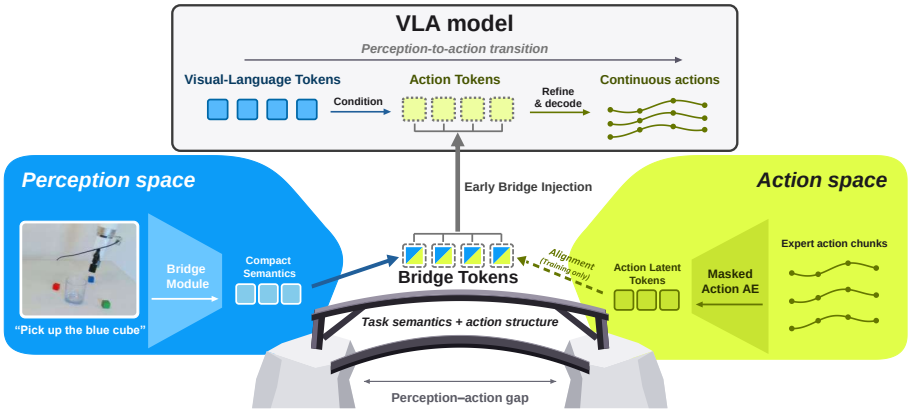


Figure 1: Conceptual view of the perception-to-action transition. Standard VLAs require Action Tokens to acquire and organize action-relevant information during decoding. PAIR instead learns perception-derived Bridge Tokens shaped by the structure of expert actions and injects them into the initial Action Tokens, so that the Action Expert begins from an action-ready intermediate state and refines it toward continuous control. The Action Autoencoder is used only during training.

explicit can facilitate action generation. Yet robot demonstrations already pair the two sides of the transition: visual–language context specifies what behavior is required, while expert actions reveal how that behavior is executed. This raises a central question:

Can we learn the missing intermediate representation directly from perception and action—a bridge that carries task semantics from one space and executable structure from the other?

We introduce PAIR, a framework for learning a *Perception–Action Intermediate Representation* directly from paired robot demonstrations. From the action side, a Masked Action Autoencoder encodes expert action chunks into Action Latent Tokens. Reconstructing the demonstrated actions requires these tokens to preserve the temporal and executable structure. From the perception side, a Bridge Module condenses the current visual-language representations into compact, task-relevant features. Aligning these features with the Action Latent Tokens transfers action structure into the perception-derived representation, forming Bridge Tokens that jointly encode task semantics and action information. Perception supplies the semantic content, expert actions supply the executable structure, and alignment integrates both into a unified bridge.

PAIR then projects the Bridge Tokens into the action-token space and injects them at the beginning of action decoding. This early Bridge Injection primes the action tokens with task-aware and action-structured information, providing an explicit starting point for the perception-to-action transition. The policy can then refine this prior through its original action-decoding process rather than constructing an action-ready representation from the final action objective alone.

Experiments on LIBERO (Liu et al., 2023), LIBERO-Plus (Fei et al., 2026), and CALVIN ABC→D (Mees et al., 2022) demonstrate consistent gains across OpenVLA-OFT and VLA-Adapter. PAIR raises the average success rate of VLA-Adapter from 97.3% to 98.4% on LIBERO and from 59.1% to 64.2% on LIBERO-Plus. On CALVIN, it improves the average completed sequence length from 4.28 to 4.40 with OpenVLA-OFT and from 4.42 to 4.53 with VLA-Adapter. Across seven real-world tasks, PAIR further raises the average success rate of OpenVLA-OFT from 51.4% to 65.0%. These improvements span standard manipulation, zero-shot distribution shifts and long-horizon execution. Representation analyses further show that Bridge Tokens retain task context while exposing action structure before decoding, supporting their role as an explicit interface between perception and action.

Contributions. Our contributions are as follows:

- We characterize the perception–action representation gap in continuous-action VLAs as the implicit transition from visual–language semantics to executable action semantics under final action supervision.

- We introduce PAIR, which learns perception-derived Bridge Tokens whose task semantics are organized by expert-action structure, and uses early Bridge Injection to provide Action Tokens with an action-aware starting point for refinement.
- We demonstrate consistent gains across standard manipulation, zero-shot distribution shifts, long-horizon execution, and real-world control, together with representation-level evidence that Bridge Tokens retain task semantics while exposing action structure before Action Expert refinement.

2 Related Work

Intermediate representations for perception-to-action learning. Intermediate representations make control-relevant structure explicit before action prediction. Motion-oriented approaches encode trajectories, visual traces, or future scene changes (Niu et al., 2025; Zheng et al., 2025; Zhang et al., 2025). Spatial representations introduce geometric information into perception and action modeling (Qu et al., 2025; Li et al., 2025), while affordance-based methods describe possible interactions or key robot configurations (Nasiriany et al., 2024; Yu et al., 2026). These approaches define the intermediate representation through a chosen physical abstraction, such as motion, geometry, or affordance. PAIR does not prescribe a particular physical meaning for the bridge. Its content is learned from the two spaces it is meant to connect: visual–language context provides task semantics, while expert actions organize the representation around executable structure. In this way, task context and action structure jointly determine what the intermediate representation retains.

Latent action representations and tokenization. Latent action methods learn compact representations of robot behavior for pretraining, prediction, and control. LAPA derives discrete latent actions from visual transitions for VLA pretraining without robot action labels (Ye et al., 2025), while UniVLA learns task-centric latent actions for policy learning (Bu et al., 2025). FAST and VQ-VLA tokenize continuous action chunks for token-based policies (Pertsch et al., 2025; Wang et al., 2025a). These approaches primarily structure how actions themselves are represented, tokenized, or predicted. LARA (Liu et al., 2026b) moves closer to the perception–action interface by aligning latent visual dynamics with intermediate Action Expert features during generation. PAIR targets a different object: a perception-derived intermediate representation that is shaped by expert-action structure during training and made available before Action Expert refinement begins.

3 Method

We consider a robot demonstration consisting of a visual observation o, a language instruction ℓ, and an H-step expert action chunk $\mathbf{A} \in \mathbb{R}^{H \times d_a}$. The VLA backbone encodes the observation and instruction into visual–language representations $\mathbf{H}_{VL} = f_{VL}(o, \ell)$, which condition an Action Expert to predict continuous actions.

A useful representation between perception and action should carry two complementary forms of information: the task semantics that determine what behavior is required, and the executable structure that determines how the behavior should unfold. As illustrated in Figure 2, PAIR learns these properties from the two sides of the transition. A Masked Action Autoencoder captures action structure from expert trajectories, while a Bridge Module extracts compact task-relevant information from the visual–language context. Aligning their horizon-wise representations forms Bridge Tokens, which are injected into Action Tokens at the beginning of the Action Expert.

At inference, the Action Autoencoder is removed and the Bridge Module predicts the task-conditioned, horizon-wise tokens directly from the current observation and instruction.

3.1 Masked Action Autoencoder

We first construct the action side of the intermediate representation. The desired representation should preserve the information that makes an expert trajectory executable, including its control content and temporal organization. We use trajectory reconstruction as a functional learning objective: a latent sequence that can recover the complete action chunk must retain the structure needed to reproduce that behavior.

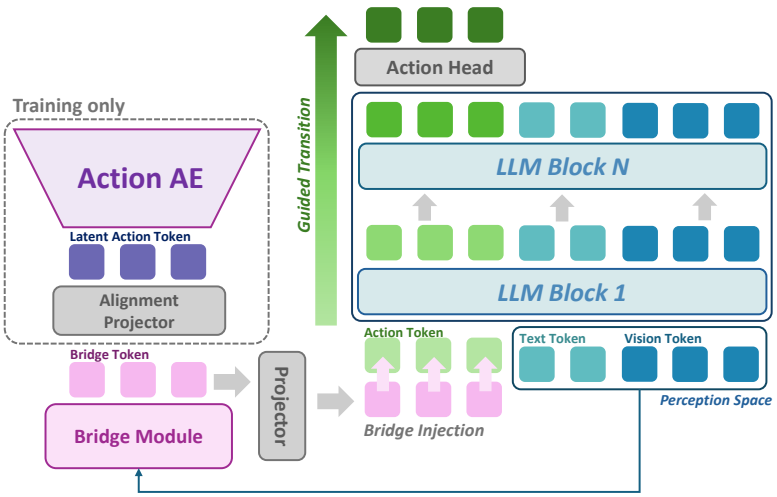


Figure 2: Overview of PAIR. The Masked Action Autoencoder encodes expert actions into Action Latent Tokens, which guide the perception-derived Bridge Tokens through representation alignment. The Bridge Tokens are projected into the Action Token space and injected at the entrance of the Action Expert.

The Masked Action Autoencoder receives a corrupted expert trajectory. The corruption operator $\mathcal{C}$ randomly masks action steps and perturbs the sequence with Gaussian noise. This corruption prevents the autoencoder from trivially copying individual action values: masking forces it to recover missing steps from temporal dependencies among action steps, while noise encourages a robust latent representation of the underlying action structure. Its encoder uses self-attention across action steps to model temporal dependencies within the action chunk. The resulting latent sequence is decoded back into the clean trajectory:

$$\widetilde{\mathbf{A}} = \mathcal{C}(\mathbf{A}), \qquad \widetilde{\mathbf{Z}}_A = E_A(\widetilde{\mathbf{A}}), \qquad \widehat{\mathbf{A}}_{\mathrm{AE}} = D_A(\widetilde{\mathbf{Z}}_A). \tag{1}$$

Here, $\widetilde{\mathbf{Z}}_A = [\widetilde{\mathbf{z}}_{A,1}, \ldots, \widetilde{\mathbf{z}}_{A,H}] \in \mathbb{R}^{H \times d_z}$ contains one Action Latent Token for each step of the action horizon.

The autoencoder is trained with an ℓ_1 reconstruction objective:

$$\mathcal{L}_{\mathrm{AE}} = \frac{1}{H} \sum_{i=1}^{H} \left\| \widehat{\mathbf{a}}_{\mathrm{AE},i} - \mathbf{a}_i \right\|_1 . \tag{2}$$

After this stage, we freeze the autoencoder. During PAIR training, its encoder processes the clean expert action chunk and produces the reference tokens $\mathbf{Z}_A = \mathrm{sg}[E_A(\mathbf{A})]$.

3.2 Bridge Module

The Action Latent Tokens encode the temporal and control structure of the demonstrated behavior. The visual–language context contains rich scene and task information. The Bridge Module uses learnable queries to extract and organize the subset most relevant to the upcoming action.

The module starts from H learnable queries $\mathbf{Q}_B = [\mathbf{q}_1, \ldots, \mathbf{q}_H]$. Through cross-attention, the queries gather task-relevant information from $\mathbf{H}_{VL}$. Self-attention across the resulting slots coordinates this information over the action horizon:

$$\mathbf{B} = G_\phi\left(\mathbf{Q}_B, \mathbf{H}_{VL}\right) \in \mathbb{R}^{H \times d_b}. \tag{3}$$

The learnable queries convert the dense visual–language context into a compact sequence with one slot for each step of the action horizon. At this stage, $\mathbf{B}$ contains task information extracted from the current observation and instruction.

Cross-space alignment. The two paths now provide horizon-aligned representations with complementary content: $\mathbf{B}$ captures the task context, while $\mathbf{Z}_A$ captures the structure of the corresponding expert behavior. We connect them through an Alignment Projector and align corresponding temporal slots:

$$\mathcal{L}_{\text{align}} = \frac{1}{H} \sum_{i=1}^{H} \left[1 - \cos\left(P_{\text{align}}(\mathbf{b}_i), \mathbf{z}_{A,i}\right)\right]. \tag{4}$$

We use the pre-projector sequence $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_H]$ as the Bridge Tokens. Their content is selected from the current visual–language context, while the alignment objective organizes that content according to expert action structure. The resulting representation therefore carries task semantics and executable information within the same horizon-wise token sequence.

3.3 Bridge Injection

Bridge Tokens are learned to mediate the transition from visual–language semantics to executable action. We place them at the beginning of the Action Expert, where Action Tokens enter the computation that develops their action semantics. This early injection primes the Action Tokens with the information represented on both sides of the transition.

Let $\mathbf{X}_A^0 \in \mathbb{R}^{H \times d_h}$ denote the Action Tokens entering the Action Expert. A projector maps the Bridge Tokens into the same hidden space, while a token-wise gate regulates their contribution across the action horizon:

$$\widetilde{\mathbf{X}}_A^0 = \mathbf{X}_A^0 + \tanh(G_g(\mathbf{B})) \odot P(\mathbf{B}). \tag{5}$$

Here, P projects the Bridge Tokens into the Action Token space, and G_g predicts an input-dependent injection weight for each temporal slot.

The Action Expert processes the primed tokens through its original computation and produces the continuous action chunk. Because task-relevant context and action structure are available from the first expert block, the Action Tokens begin from an explicit intermediate state and refine it toward executable control.

3.4 Training

The two training stages follow the construction of the intermediate representation. The first stage learns action structure by optimizing the Masked Action Autoencoder with $\mathcal{L}_{\text{AE}}$. The second stage freezes the autoencoder and transfers this structure to the perception-derived Bridge Tokens while learning to use them inside the policy.

Let $\mathcal{L}_{\text{act}}$ denote the original continuous action prediction loss of the VLA policy. The second-stage objective is

$$\mathcal{L}_{\text{PAIR}} = \mathcal{L}_{\text{act}} + \lambda_{\text{align}} \mathcal{L}_{\text{align}}, \tag{6}$$

where λ_{align} controls the contribution of the representation alignment objective. The VLA policy and bridge module are optimized jointly in this stage.

During inference, the Bridge Module extracts Bridge Tokens from the current visual–language representations. Bridge Injection then primes the Action Tokens at the beginning of the Action Expert for continuous action prediction.

4 Experiments

Our experiments address three questions:

- Does PAIR improve manipulation performance, zero-shot generalization, and long-horizon execution, and do these gains extend to real-world tasks? (Secs. 4.2–4.3)
- Do Bridge Tokens encode task semantics and action structure before the Action Expert, and how does their action information compare with baseline Action Tokens across layers? (Section 4.4)
- How do action alignment and injection position affect PAIR's performance, and what are the contributions of adaptive injection and autoencoder noise? (Section 4.5; Appendix A)

4.1 Experimental Setup

Benchmarks and metrics. LIBERO (Liu et al., 2023) evaluates task-conditioned manipulation across the Spatial, Object, Goal, and Long suites, which vary spatial relationships, manipulated objects, task goals, and action-sequence length. We evaluate each task over 50 rollouts from the official initial states and report success rates for each suite and their average. Following the official protocol, LIBERO-Plus (Fei et al., 2026) evaluates the corresponding LIBERO-trained checkpoints zero-shot under changes in camera viewpoint, robot initial state, language instruction, lighting, background, sensor noise, and object layout. CALVIN ABC→D (Mees et al., 2022) trains policies in environments A–C and evaluates 1,000 five-task chains in unseen environment D. We report success at completing at least one to five consecutive tasks and average completed sequence length, capturing both individual-task performance and sustained execution. Additional protocols are in Appendix C.

Backbones and implementation. We evaluate PAIR with VLA-Adapter (Wang et al., 2026) on LIBERO/Plus, both VLA-Adapter and OpenVLA-OFT (Kim et al., 2025a) on CALVIN, and OpenVLA-OFT on the real robot. Each **+ PAIR** row augments the preceding backbone. We train on four NVIDIA RTX 6000 Ada Generation GPUs; detailed training settings are provided in Appendix C.

4.2 Simulation Results

We report PAIR's aggregate results as mean ± standard deviation over three random seeds.

Table 1: Simulation results on LIBERO. Success rate (%) is reported for each task suite and their average. Best results are in bold.

Method	Spatial	Object	Goal	Long	Avg.
SpatialVLA (Qu et al., 2025)	88.2	89.9	78.6	55.5	78.1
WorldVLA (Cen et al., 2025)	87.6	96.2	83.4	60.0	81.8
π_0-FAST (Pertsch et al., 2025)	96.4	96.8	88.6	60.2	85.5
LARA (full) (Liu et al., 2026b)	88.0	92.0	88.5	86.0	88.6
GR00T N1 (Bjorck et al., 2025)	94.4	97.6	93.0	90.6	93.9
π_0 (Black et al., 2025)	96.8	98.8	95.8	85.2	94.2
UniVLA (Bu et al., 2025)	96.5	96.8	95.6	92.0	95.2
UnifiedVLA (Wang et al., 2025b)	95.4	98.8	93.6	94.0	95.5
OpenVLA-OFT (Kim et al., 2025a)	97.6	98.4	97.9	94.5	97.1
$\pi_{0.5}$-LARA (Liu et al., 2026b)	99.0	98.5	99.0	94.5	97.8
VLA-Adapter (Wang et al., 2026)	97.8	99.2	97.2	95.0	97.3
+ PAIR	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**

LIBERO: task-conditioned manipulation. PAIR raises VLA-Adapter's average success from 97.3% to $(98.4 \pm 0.3)\%$, improving all four suites (Table 1). Gains of 1.4 points on Spatial and Goal and 1.2 on Long extend the benefit of the bridge from basic manipulation to longer action sequences. The Object suite also improves from 99.2% to 99.6%.

Table 2: Zero-shot LIBERO-Plus results with LIBERO-trained checkpoints. Total follows the benchmark's overall score, not the seven-category mean. Best results are in bold.

Method	Camera	Robot	Language	Light	Background	Noise	Layout	Total
OpenVLA (Kim et al., 2025b)	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
WorldVLA (Cen et al., 2025)	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0
NORA (Hung et al., 2025)	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
UniVLA (Bu et al., 2025)	1.8	**46.2**	69.6	69.0	81.0	21.2	31.9	42.9
π_0 (Black et al., 2025)	13.8	6.0	58.8	**85.0**	**81.4**	**79.0**	68.9	53.6
VLA-Adapter (Wang et al., 2026)	36.2	37.9	74.6	70.6	76.1	58.0	69.7	59.1
+ PAIR	**44.3**	45.7	**75.1**	75.8	80.3	64.7	**72.6**	**64.2**

LIBERO-Plus: zero-shot OOD generalization. PAIR improves all seven OOD categories, increasing overall success from 59.1% to $(64.2 \pm 0.4)\%$ (Table 2). The largest gains occur for camera

viewpoint (+8.1 points), robot initial state (+7.8), and noise (+6.7), followed by lighting (+5.2), background (+4.2), and layout (+2.9); language perturbations improve by 0.5 points. Comparing the same LIBERO-trained checkpoints across both benchmarks, PAIR gains 1.1 points on LIBERO and 5.1 points under distribution shift. The overall success-rate drop from LIBERO to LIBERO-Plus decreases from 38.2 points for VLA-Adapter to 34.2 points with PAIR.

CALVIN: long-horizon execution. PAIR improves sustained multi-task execution with both backbones (Table 3). Average sequence length increases from 4.28 to 4.40 ± 0.04 for OpenVLA-OFT and from 4.42 to 4.53 ± 0.03 for VLA-Adapter; five-task success rises by 3.9 and 3.7 points, respectively. For comparison, the gains in single-task success are 0.7 and 0.1 points. For both backbones, the margin over the baseline grows as more consecutive tasks are required, showing that the benefit is more pronounced in sustained execution than in completing the first task alone.

Table 3: Simulation results on CALVIN ABC→D. We report success rates (%) for completing at least one to five consecutive tasks and average completed sequence length. Best results are in bold.

Method	1	2	3	4	5	Avg. Len.
OpenVLA (Kim et al., 2025b)	91.3	77.8	62.0	52.1	43.5	3.27
RoboVLMs (Li et al., 2026)	98.0	93.6	85.4	77.8	70.4	4.25
VPP (Hu et al., 2025)	96.5	90.9	86.6	82.0	76.9	4.33
UnifiedVLA (Wang et al., 2025b)	98.9	94.8	89.0	82.8	75.1	4.41
DreamVLA (Zhang et al., 2025)	98.2	94.6	89.5	83.4	78.1	4.44
NIAF (Liu et al., 2026a)	99.7	95.9	90.6	84.8	76.4	4.47
HARP-VLA (Zhu et al., 2026)	**99.8**	**96.7**	91.3	84.4	75.9	4.48
VLA-Adapter-Pro (Wang et al., 2026)	98.5	95.0	90.5	85.3	80.0	4.50
LIRA (Zhang et al., 2026)	98.8	95.3	91.0	85.8	**80.5**	4.52
OpenVLA-OFT (Kim et al., 2025a)	96.9	92.0	85.7	80.4	72.9	4.28
+ PAIR	97.6	93.1	88.8	83.7	76.8	4.40
VLA-Adapter (Wang et al., 2026)	99.1	94.6	88.8	82.8	76.5	4.42
+ PAIR	99.2	96.1	**91.7**	**86.0**	80.2	**4.53**

4.3 Real-World Experiments

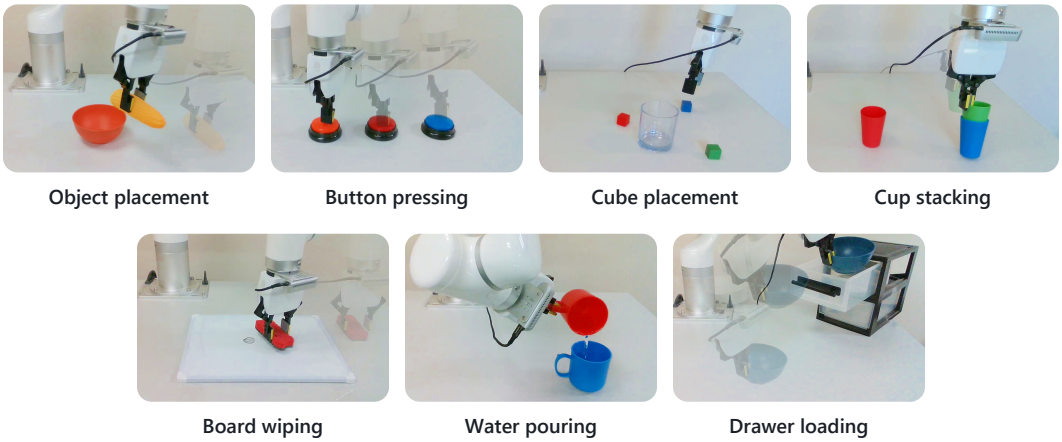


Figure 3: Seven real-world tasks on the UF850, spanning language grounding, multi-step execution, and precise manipulation.

Platform and tasks. We evaluate seven tasks on a UFACTORY UF850 with synchronized third-person and wrist-mounted Intel RealSense RGB observations and robot proprioception (Figure 3). Object placement, cube placement, and cup stacking require selecting or arranging objects according to the instruction; button pressing requires executing a specified color order. Board wiping involves sustained tool contact, water pouring requires controlled liquid transfer, and drawer loading combines opening the specified drawer with placing a bowl inside. Together, these tasks cover

language-conditioned selection, sequential interaction, precise placement, and contact-rich manipulation. Task instructions and demonstration sequences are provided in Appendix C.3.

Demonstration data and evaluation. We collect expert demonstrations through Quest VR teleoperation, with 30–51 trajectories per task and 293 trajectories in total. OpenVLA-OFT and PAIR use identical processed demonstrations and train a separate policy for each task. Each method is evaluated over 20 trials per task, totaling 140 trials across the seven tasks.

Results. PAIR improves OpenVLA-OFT on all seven tasks, from 51.4% to 65.0% overall (Figure 4). Gains span object placement (+15 points), sequential button pressing (+5), and drawer loading (+15), with the largest gain in water pouring (+20). Improvements therefore appear in both instruction-driven discrete interactions and tasks requiring extended control over object motion or contact, extending the simulation results to physical manipulation.

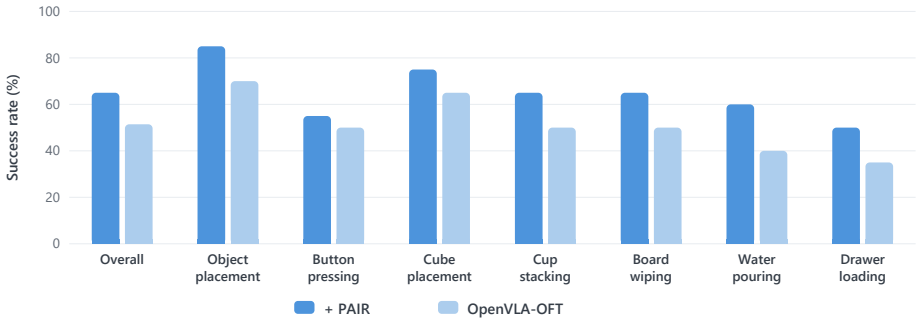


Figure 4: Real-world success rates over 20 trials per task. PAIR improves OpenVLA-OFT across all seven tasks, with an overall gain of 13.6 percentage points.

4.4 Analysis

Using frozen representations from 500 task-balanced LIBERO-Spatial frames, Figure 5(a) uses five-fold linear probes with trajectory-grouped splits to predict ten task identities and eight coarse action modes clustered from ground-truth action chunks. It compares Bridge Tokens with latents from the action autoencoder. Figure 5(b) decodes the full eight-step, seven-dimensional action chunk from pre-Transformer Bridge Tokens and baseline Action Tokens after each block. Both panels use the same 500 frames and trajectory-grouped folds. Higher accuracy or R^2 means more linearly accessible information; details of the layerwise action-decoding and geometry analyses are provided in Appendix B.

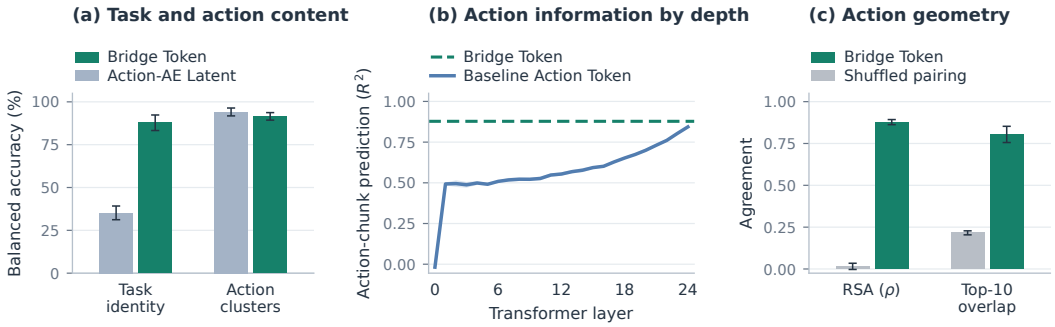


Figure 5: **Bridge Tokens connect task semantics to action structure.** (a) Linear-probe balanced accuracy for task identity and action clusters. (b) Action-chunk R^2 for baseline Action Tokens across layers (blue) and Bridge Tokens before the Action Expert (dashed). (c) RSA and top-10 neighbor overlap with ground-truth actions. Error bars show fold standard deviations in (a,b), and task standard deviations for Bridge Tokens or 5th–95th permutation percentiles for shuffled controls in (c).

Bridge Tokens encode both task and action information. Figure 5(a) contrasts task identity with coarse action-mode decodability. Bridge Tokens achieve $87.8\% \pm 4.5\%$ task accuracy versus $35.2\% \pm 4.0\%$ for action-only autoencoder latents. They also retain strong action-cluster accuracy ($91.5\% \pm 2.2\%$), compared with $94.1\% \pm 2.3\%$ for the action-only representation. Bridge Tokens therefore combine substantially richer task information with strongly accessible action structure.

Bridge Tokens make actions readable before Action Expert refinement. Baseline Action Tokens progressively acquire continuous-action information through the Transformer, reaching $R^2 = 0.844$ only at the final block. Bridge Tokens already reach $R^2 = 0.878$ before the first block (Figure 5(b)). Action information is therefore linearly accessible in Bridge Tokens at the entrance of the Action Expert.

The bridge preserves relationships between actions. Figure 5(c) compares Bridge-token and ground-truth action geometry using cross-trajectory pairs within each task. RSA reaches 0.878 and top-10 neighbor overlap reaches 80.4%, versus 0.014 and 21.6% after shuffling. Bridge Tokens therefore preserve continuous action geometry beyond task identity.

4.5 Ablation Studies

We ablate the injection position and alignment weight on LIBERO with VLA-Adapter. Each setting is evaluated across all four suites. The action-side target ablation appears in Appendix A.

Table 4: Injection position ablation on the four LIBERO task suites. Values are success rates (%).

Position	Spatial	Object	Goal	Long	Avg.
Start (base)	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**
End	98.4	99.0	97.6	94.6	97.4
Middle	98.8	98.9	98.3	95.6	97.9
All	98.7	99.0	97.9	94.8	97.6

An early bridge is more effective than repeated injection. Entrance injection achieves 98.4% average success, above end (97.4%), middle (97.9%), and all-layer (97.6%) injection (Table 4). This supports an action-informed starting point for refinement (Section 4.4).

Table 5: Alignment loss weight ablation on the four LIBERO task suites. Values are success rates (%).

λ_{align}	Spatial	Object	Goal	Long	Avg.
0.1 (base)	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**
0.01	98.8	99.2	98.0	95.2	97.8
0	98.2	98.8	97.4	94.4	97.2

Action alignment strengthens the bridge. Without alignment, average success is 97.2%; $\lambda_{\text{align}} = 0.01$ and 0.1 reach 97.8% and 98.4% (Table 5). Alignment shapes bridge content, while entrance injection sets when it becomes available to the Action Expert.

5 Conclusion

PAIR introduces an explicit intermediate representation for the perception-to-action transition in continuous-action VLAs. Bridge Tokens combine visual–language context with action structure and prime Action Tokens at the entrance of the Action Expert. Across LIBERO, LIBERO-Plus, CALVIN, and seven real-world tasks, PAIR improves two VLA backbones in standard manipulation, zero-shot generalization, long-horizon execution, and physical control. Representation analyses show that the learned bridge carries both task and action information, while ablations identify action alignment and early injection as its key ingredients. These results establish explicit perception–action intermediates as a practical design principle for VLA policies.

AI USE STATEMENT

Large language models were used to aid language polishing and improve the clarity of the manuscript. All scientific ideas, experimental design, implementation, analysis, results, and conclusions were produced and verified by the authors. We reviewed all AI-assisted text and take responsibility for the final content of this work.

REFERENCES

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, et al. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to act anywhere with task-centric latent actions. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.014.

Jun Cen, Chaohui Yu, Hangjie Yuan, et al. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Senyu Fei, Siyin Wang, Junhao Shi, et al. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, 2026.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, et al. Video prediction policy: A generalist robot policy with predictive visual representations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24328–24346. PMLR, 2025.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. NORA: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b.

Peiyan Li, Yixiang Chen, Hongtao Wu, et al. BridgeVLA: Input-output alignment for efficient 3D manipulation learning with vision-language models. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Xinghang Li, Peiyan Li, Long Qian, et al. What matters in building vision-language-action models for generalist robots. *Nature Machine Intelligence*, 8:158–172, 2026. doi: 10.1038/s42256-025-01168-7.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

Haoyun Liu, Jianzhuang Zhao, Xinyuan Chang, et al. Neural implicit action fields: From discrete waypoints to continuous functions for vision-language-action models. *arXiv preprint arXiv:2603.01766*, 2026a.

Mengya Liu, Baoxiong Jia, Jiangyong Huang, Jingze Zhang, and Siyuan Huang. LARA: Latent action representation alignment for vision-language-action models. *arXiv preprint arXiv:2606.07100*, 2026b.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Soroush Nasiriany, Sean Kirmani, Tianli Ding, Laura Smith, Yuke Zhu, Danny Driess, Dorsa Sadigh, and Ted Xiao. RT-Affordance: Affordances are versatile intermediate representations for robot manipulation. *arXiv preprint arXiv:2411.02704*, 2024.

Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. LLARVA: Vision-action instruction tuning enhances robot learning. In *Conference on Robot Learning*, volume 270, pp. 3333–3355, 2025.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, et al. FAST: Efficient action tokenization for vision-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.012.

Delin Qu, Haoming Song, Qizhi Chen, et al. SpatialVLA: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011.

Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. VQ-VLA: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11089–11099, 2025a.

Yihao Wang, Pengxiang Ding, Lingxiao Li, et al. VLA-Adapter: An effective paradigm for tiny-scale vision-language-action model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(22):18638–18646, 2026. doi: 10.1609/aaai.v40i22.38931.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025b.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, et al. Latent action pretraining from videos. In *International Conference on Learning Representations*, 2025.

Qize Yu, Jiadi You, Yuran Wang, et al. AffordanceVLA: A vision-language-action model empowering action generation through affordance-aware understanding. *arXiv preprint arXiv:2606.06155*, 2026.

Wenyao Zhang, Hongsi Liu, Zekun Qi, et al. DreamVLA: A vision-language-action model dreamed with comprehensive world knowledge. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zhewei Zhang, Puyue Wang, Guanren Qiao, et al. LIRA: Local cross-layer information routing for vision-language-action decoding. *arXiv preprint arXiv:2608.07596*, 2026.

Qingqing Zhao, Yao Lu, Moo Jin Kim, et al. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713, 2025.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé, III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *International Conference on Learning Representations*, 2025.

Xiang Zhu, Puzhen Yuan, Yichen Liu, and Jianyu Chen. HARP-VLA: Human-robot aligned representation learning for vision-language-action model. *arXiv preprint arXiv:2605.31234*, 2026.

A Additional Ablations

We report further comparisons on the four LIBERO suites using VLA-Adapter. The base setting is the same as in the main-text ablations; all table entries are success rates (%).

Table 6: Action-side alignment target ablation on LIBERO. Success rates (%).

Alignment Target	Spatial	Object	Goal	Long	Avg.
Raw expert actions	98.2	98.6	97.0	94.4	97.1
Action Latent Tokens (base)	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**

Learned action tokens are a stronger alignment target. The raw-action variant aligns the bridge directly to expert action chunks, whereas PAIR uses the Action Autoencoder's horizon-aligned Action Latent Tokens. The latter reaches 98.4% average success versus 97.1% for raw actions (Table 6), improving all four suites; its largest gain is 1.8 points on Long.

Table 7: Injection strength ablation on the four LIBERO task suites. Values are success rates (%).

Injection Strength	Spatial	Object	Goal	Long	Avg.
Learnable (base)	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**
1.0	98.8	99.4	98.2	95.6	98.0
0.5	98.9	99.3	98.1	95.9	98.1

Adaptive weighting improves over a fixed contribution. Learnable injection reaches 98.4%, versus 98.0% and 98.1% for fixed strengths of 1.0 and 0.5 (Table 7), supporting adaptive bridge weighting.

Table 8: Action AE noise ablation on the four LIBERO task suites. Values are success rates (%).

AE Training	Spatial	Object	Goal	Long	Avg.
Noise (base)	**99.2**	**99.6**	**98.6**	**96.2**	**98.4**
Clean	98.4	99.0	97.6	95.4	97.6

Noise during autoencoder training. Training with noise improves average success from 97.6% to 98.4% and improves every suite. Denoising therefore produces action structure that remains useful under input corruption.

B Additional Analysis

B.1 Layerwise Probe Protocol

For the layerwise analysis in Figure 6, we use frozen representations from LIBERO-Spatial, with 500 training-set frames balanced across the ten tasks. Each example contains eight pre-LLM Bridge Tokens, the corresponding Action Latent Tokens, and an eight-step, seven-dimensional ground-truth action chunk. Both the Bridge–Action Latent and Bridge–baseline Action Token comparisons use the same 500 frames, action targets, and trajectory-grouped folds. Baseline VLA-Adapter Action Tokens are extracted before the Transformer and after each of its 24 blocks.

Classification and regression. Representations are flattened, standardized, and reduced to 50 principal components when their input dimension exceeds 128. Preprocessing is fit on the training portion of each fold. Task classification uses the ten task identities; action classification uses eight KMeans clusters of standardized ground-truth action chunks ($K = 8$, 20 initializations, seed 7). A multinomial logistic-regression probe with $C = 1$ is evaluated using five trajectory-grouped folds, with all frames from each trajectory assigned to the same fold. We report mean balanced accuracy and fold standard deviation. Continuous-action prediction uses ridge regression ($\alpha = 10$) on the same trajectory-grouped folds with training-fold target standardization; we report variance-weighted R^2. The split seed is 7.

B.2 Task and Action Information Across Depth

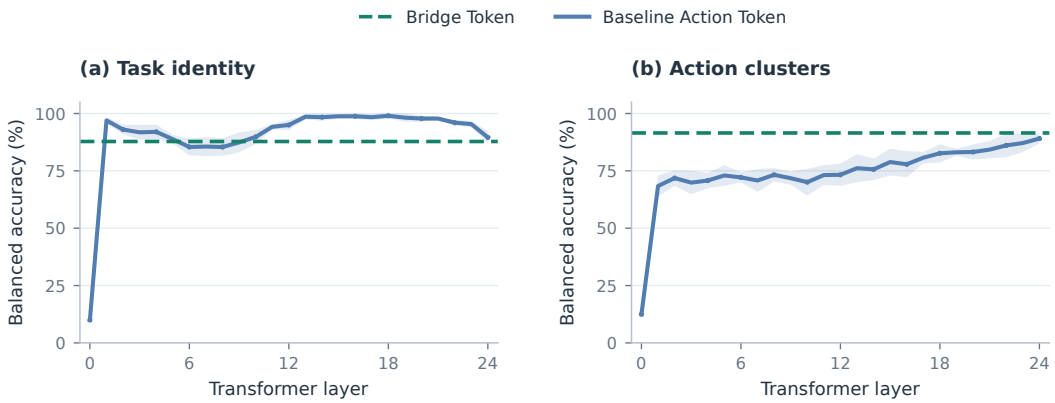


Figure 6: **Task identity and action-cluster decodability across depth.** Blue curves show baseline Action Tokens with fold-standard-deviation bands. Green dashed lines indicate the pre-LLM Bridge Token score. Layer 0 precedes the Transformer; layers 1–24 denote completed blocks.

Figure 6 separates access to task identity from access to action patterns. Baseline Action Tokens acquire task information rapidly: task accuracy reaches 97.0% after the first block and remains high through much of the network, ending at 89.6%. Their action-cluster accuracy develops more gradually and reaches 89.1% at the final block. Bridge Tokens provide 87.8% task accuracy together with 91.5% action-cluster accuracy before entering the Action Expert, making both forms of information accessible from the beginning of action generation.

Depth-wise transition. The two baseline curves evolve differently: task identity becomes accessible after the first block, whereas action-cluster structure accumulates more gradually. The Bridge Token operating point combines both signals before any Action Expert block. Its task accuracy remains close to the baseline's late-layer level, while its action-cluster accuracy already exceeds the final baseline value. This pattern complements the entrance-injection ablation: PAIR makes an action-aware state available at the start of refinement, while the baseline develops action structure across later blocks.

B.3 Continuous-Action Decoding Across Depth

Figure 7 directly compares how much continuous control information is readable from the two Action-Token pathways. After Bridge Token injection, PAIR begins at 47.3% decoding score, compared with 20.1% for the baseline tokens before the Transformer. The advantage is largest at the entrance, remains visible across most early and middle depths, and reappears at the final block (76.4% versus 72.7%). Thus, the injected Action Tokens start from a more action-readable state and retain that advantage through refinement.

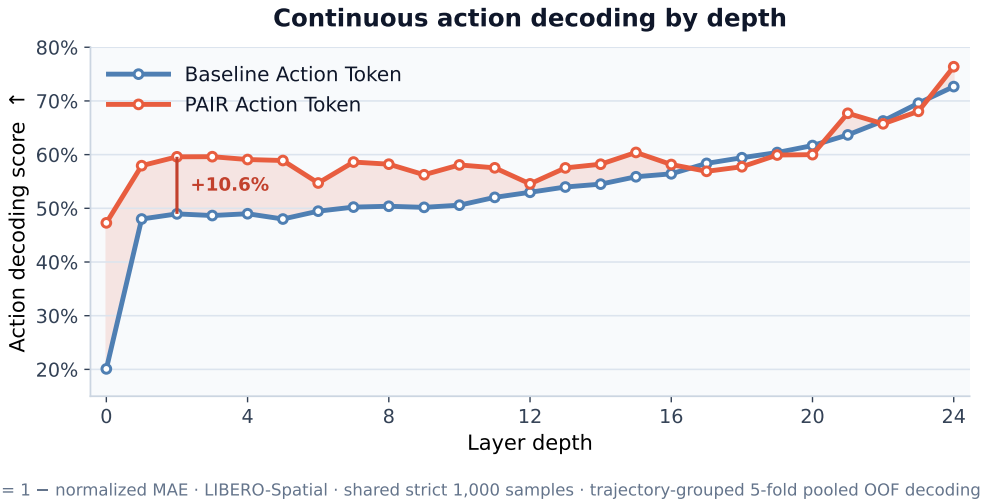


Figure 7: **Continuous-action decodability of Action Tokens across depth.** Ridge readouts predict the full eight-step ground-truth action chunk from frozen Action Tokens. Scores are $1 - \mathrm{MAE}$ after target normalization, pooled over five trajectory-grouped held-out folds on 1,000 shared LIBERO-Spatial examples. PAIR tokens are measured after Bridge Token injection; baseline tokens are measured before and after each Transformer block.

B.4 Additional Action-Geometry Evidence

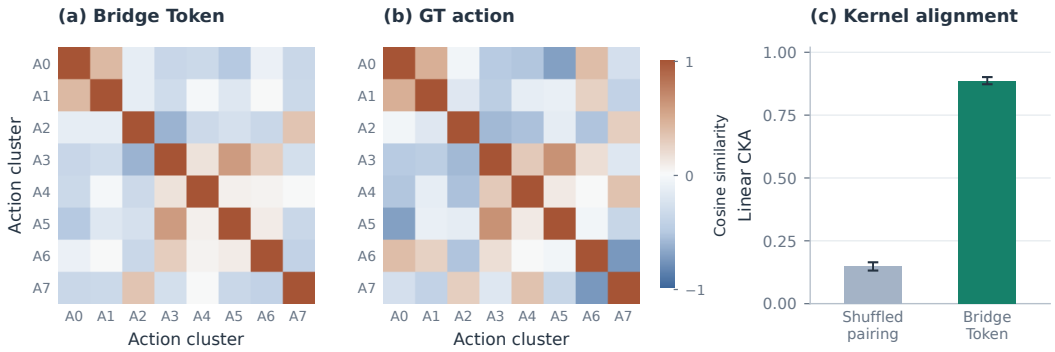


Figure 8: **Action-cluster geometry and kernel alignment.** (a,b) Cosine similarities between GT-derived action-cluster centroids in the two spaces, using a shared color scale. A0–A7 index unsupervised clusters rather than named behaviors. (c) Within-task linear CKA. Error bars indicate task standard deviation for Bridge Tokens and the 5th–95th percentile interval of the trajectory-permutation reference.

Sampling and comparison. The geometry analysis uses a separate deterministic 500-frame subset: two ordered frames from each of 25 trajectories per task. Bridge representations and ground-truth actions are independently flattened, feature-wise standardized, centered within each task, and L2-normalized. RSA compares cosine-distance ranks for same-task, cross-trajectory pairs. Top-10 neighbor overlap uses the same task and trajectory restrictions. Linear CKA compares the centered cosine Gram matrices within each task. Scores are averaged across the ten tasks. The permutation reference reassigns complete two-frame trajectory groups within each task 200 times while retaining their internal order.

Action geometry across complementary metrics. The centroid heatmaps summarize relationships among action-cluster means. Positive similarity between A0 and A1 and between A3 and A5, and negative similarity between A2 and A3, are retained across the two spaces. Differences between the matrices reflect the perceptual and task context carried by Bridge Tokens. At the sample level, linear CKA reaches 0.887 compared with 0.148 under shuffled correspondence, with task-wise values ranging from 0.865 to 0.909. Together with the distance-ranking and neighborhood results in the main text, these metrics show action structure at both sample and cluster levels.

C Implementation and Evaluation Details

C.1 Training Hyperparameters

Table 9: Default training hyperparameters. Batch denotes the effective batch size across four GPUs. The OpenVLA-OFT learning rate remains constant because its decay milestone follows the 50k-step run.

Stage / backbone	Batch	Steps	Learning rate	Schedule	LoRA rank
Action Autoencoder	64	50k	3×10^{-4}	cosine, 1k warmup	–
VLA-Adapter + PAIR	64	150k	2×10^{-4}	cosine	64
OpenVLA-OFT + PAIR	64	50k	5×10^{-4}	constant	32

Additional settings. The Action Autoencoder is trained for 50k steps on LIBERO demonstrations for LIBERO and zero-shot LIBERO-Plus, on CALVIN demonstrations for CALVIN, and separately on each task's demonstrations for the seven real-world tasks. We use AdamW for both stages. The Action Autoencoder uses weight decay 10^{-4} and clips the gradient norm at 1.0. Policy training uses bfloat16, image augmentation, and alignment weight $\lambda_{\text{align}} = 0.1$.

PAIR architecture. The Action Autoencoder comprises a two-layer pre-norm Transformer encoder and a one-layer pre-norm Transformer decoder, both using self-attention over the action chunk with four attention heads. During training, action steps are randomly masked with probability 0.5 and Gaussian noise with standard deviation 0.05 is added before reconstruction. Both OpenVLA-OFT and VLA-Adapter use eight Bridge Tokens, matching the eight action-step tokens at the input to their final Action Experts. The Bridge Module forms them from learned horizon-aligned queries using one pre-norm cross-attention block over the visual–language tokens, followed by one pre-norm self-attention block; both use eight attention heads and residual MLP blocks. The alignment and injection projections are two-layer MLPs with GELU activations. Injection is scaled by a zero-initialized per-token gate implemented by layer normalization, a scalar linear projection, and a $\tanh$ nonlinearity.

C.2 Simulation Benchmarks

LIBERO. The Action Autoencoder is trained on LIBERO data, and the VLA-Adapter policy is trained on the corresponding suite. LIBERO (Liu et al., 2023) evaluates task-conditioned manipulation across the Spatial, Object, Goal, and Long suites. These suites vary spatial relationships, manipulated objects, task goals, and the length of the required action sequence. We evaluate each task for 50 rollouts from the official initial states using 256×256 environment observations, and report each suite's success rate and their average.

LIBERO-Plus. Following the official evaluation protocol (Fei et al., 2026), we directly evaluate the corresponding LIBERO-trained checkpoints zero-shot. The seven perturbation categories cover camera viewpoint, robot initial state, language instruction, lighting, background, sensor noise, and object layout. Per-category and overall success assess generalization beyond the original training conditions.

CALVIN. We use the ABC→D split (Mees et al., 2022): policies train in environments A–C and execute 1,000 five-task chains in unseen environment D. OpenVLA-OFT executes each eight-step

action chunk open-loop before replanning. We report success at completing at least one to five consecutive tasks and average completed sequence length, measuring both individual-task performance and sustained execution.

C.3 Real-World Tasks and Evaluation

Platform, data, and evaluation. All physical experiments use a UFACTORY UF850 arm with synchronized wrist-mounted and fixed third-person Intel RealSense RGB cameras. The policy receives both RGB views and robot proprioception. Expert demonstrations are collected through Quest VR teleoperation and converted to RLDS. For each task, we train a separate Action Autoencoder for PAIR and separate policies for OpenVLA-OFT and PAIR on identical processed trajectories. We evaluate 20 trials per task (140 in total) and report the macro-average across the seven tasks.

Table 10: Real-world task definitions and dataset sizes. X, Y, and Z denote instance-specific objects or colors.

Task	Instruction and success criterion	Demos.
Object placement	"Place the object in the Y bowl." Place the object in the instructed bowl.	51
Button pressing	"First press X, then Y, and finally Z." Press the three buttons in the specified order.	50
Cube placement	"Put X in the cup." Select the instructed cube among the candidates and place it in the cup.	51
Cup stacking	"Stack X on top of Y." Stack the instructed source cup on the target cup.	51
Board wiping	"Wipe the marked circle off the whiteboard." Grasp the eraser and remove the marked circle.	30
Water pouring	"Pour water from the red cup into the blue cup." Transfer the water between the specified cups.	30
Drawer loading	"Open the second drawer and place the blue bowl inside." Open the specified drawer and place the bowl inside.	30

Task coverage. The task suite spans instruction-conditioned target selection in object and cube placement, relative object reasoning and terminal alignment in cup stacking, ordered execution in button pressing, and contact-rich or articulated manipulation in board wiping, water pouring, and drawer loading. Together, these tasks cover discrete task specification, sequential control, and extended physical interaction under a shared hardware and observation setup.

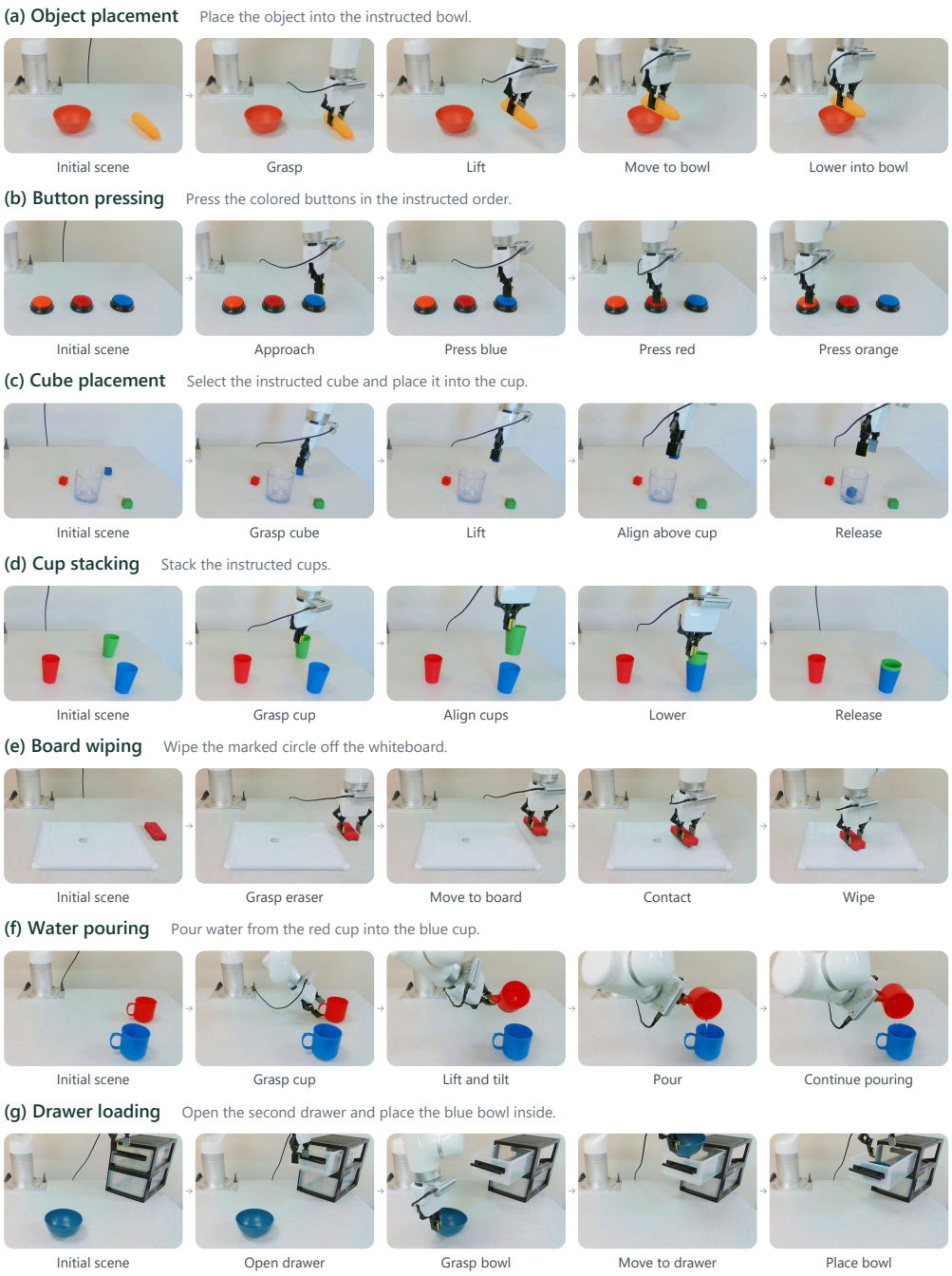


Figure 9: **Demonstration sequences for the seven real-world tasks.** Each row shows five chronological frames from one recording, with short labels identifying the manipulation stages.